Report 2047356-TN-13 (D5.1 + D5.2): Spatially 2-D plasma model incorporating velocity space effects

David Moxey, King's College London
Ben Dudson, Peter Hill, Ed Higgins, David Dickinson & Steven Wright, University of York

March 31, 2022

Contents

1 Executive summary

This report briefly outlines potential routes for implementation of the system 2-6 equations within the *Nektar++* framework. Owing to the practicality of implementing such a system within the limited resources of this project, we focus on the steps required to implement such a system, highlighting where improvements should be made within *Nektar++* to enable future implementation.

2 Introduction

As taken from the equations document supplied with NEPTUNE [1], system 2-6 is a spatially 2D plasma model incorporating velocity space effects and is expressed in the following manner for the electron number density n_e, vorticity $\nabla \cdot \mathbf{E}^+$, electron energy $\mathcal{E}_e$, ion energy $\mathcal{E}_i$ and neutral number density n_n:

$$\frac{\partial n_e}{\partial t} + \nabla \cdot (n_e \mathbf{u}_e) = S_{n_e} - \frac{n_e}{\tau_{n_e}}$$

$$\frac{\partial}{\partial t}(\nabla \cdot \mathbf{E}^+) + \nabla \cdot \left(\nabla \cdot \left(\mathbf{u}_i \otimes \mathbf{E}^+\right)\right) = \nabla \cdot \left(n_i \left(\mathbf{u}_{\nabla Bi} + \mathbf{u}_{cx}\right) - \frac{1}{Z_i} n - e\mathbf{u}_{\nabla Be}\right) + \frac{1}{Z_i}\frac{n_e}{\tau_{n_e}} - \frac{n_i}{\tau_{n_i}} + \nabla \cdot (\nu \nabla_\perp (\nabla \cdot \mathbf{E}^+))$$

$$\frac{\partial \mathcal{E}_e}{\partial t} + \nabla \cdot (\mathcal{E}_e \mathbf{u}_e + p_e \mathbf{u}_e) = S_{\mathcal{E}_e} - \frac{\mathcal{E}_e}{\tau_{Ee}} + Q_{ie} + \nabla \cdot (\chi_{\perp e} n_e \nabla_\perp T_e)$$
$$\frac{\partial \mathcal{E}_i}{\partial t} + \nabla \cdot (\mathcal{E}_i \mathbf{u}_i + p_i \mathbf{u}_i) = S_{\mathcal{E}_i} - \frac{\mathcal{E}_i}{\tau_{Ei}} + Q_{ie} + \nabla \cdot (\chi_{\perp i} n_i \nabla_\perp T_i)$$
$$\frac{\partial n_n}{\partial t} = S_{n_n} + \nabla \cdot (D_n \nabla_\perp p_n)$$

where p_α is pressure, T_α is temperature, Z_i is charge state. We omit the definitions of the various sub-terms and constants to the equation document [1] as the equation system itself provides sufficient motivation for the remainder of this document.

3 Test cases (D5.1)

D5.1 focuses on the development of adequate test cases for this set of equations. Given the inexperience of the primary author of this deliverable with the physics exhibited in these systems, we will not discuss here the consideration of simplified versions of this system under the selection of simplified parameters (although such a case may be possible).

In lieu of this, and also considering that this approach would not consider every term's correctness by definition, a sensible alternative strategy in this case would to be consider a functional test of the solver via the method of manufactured solutions (MMS) [2], which is an approach previously used successfully in e.g. BOUT++ [3], within the test suite of *Nektar++*, and in many other codes where analytic solutions are unavailable. In this setting, a solution field is chosen *a priori*, and the equations evaluated analytically using this manufactured solution. A source term is then added to the equations with these calculated terms, which then permits verification of the computed discretised equations. This permits a number of benefits, namely that:

- In the limit of small element size and increasing polynomial order, this should therefore serve as a mechanism to test the formal accuracy of the solver.
- Given an appropriate choice of manufactured solution, this yields a functional unit test of all of the terms from the equations.
- This removes the limitation of the complexity of the equations or the need for exact analytic solutions.

The MMS approach does, however, come with a number of limitations.

- The choice of domain and solution is important in order to ensure correctness is tested adequately. For example, a zero solution field is technically a 'correct' solution, but unlikely to yield any useful validation of the solver.
- For higher-order methods, the use of non-polynomial manufactured solutions is also critical in ensuring that all modes of the solution are adequately energised, and sometimes this can be difficult to ensure across the entire domain of interest.
- In parallel with the above, the complexity of the system likely necessitates the use of e.g. a computer-algebra system to compute desired source terms accurately and without human error.
- Issues due to timestepping problems will very likely not be highlighted in this approach (at least in the experience of this author).

4 Path to implementation within *Nektar++* (D5.2)

The complexity of system 2-6 poses some challenges in terms of implementation, particularly considering the existing infrastructure within *Nektar++*. In this section we highlight a few of the expected bottlenecks and potential mitigations for these.

4.1 Discretisation choice

Given the conservative form of the equations as written and need to ensure accuracy in terms of mass conservation, the anticipated discretisation would be a discontinuous Galerkin (DG) formulation, which should map naturally onto the equations at hand. With this said, there are some terms (particularly for e.g. the vorticity) which will need careful consideration in terms of the weak formulation. Additionally, certain terms such as the electrostatic potential could potentially be solved in a continuous manner for efficiency and accuracy purposes. This joint-discretisation methodology has been shown to work quite successfully in *Nektar++* for e.g. the implementation of the Hasegawa-Wakatani proxyapp, whereby the DG method acts to stabilise the equations (thereby meaning that hyperelliptic terms are not required for stability), whilst the CG method provides a route towards obtaining the elliptic solution for ϕ robustly and efficiently.

4.2 Explicit vs. implicit solutions

An initial implementation of these terms within *Nektar++* will need to be explicit in time. There are a number of reasons to motivate this:

- From a practical perspective this is the path of least resistance and the infrastructure for this is relatively robust, with several solvers (such as the compressible Navier-Stokes equations) implemented in this manner.
- The dependence of terms as a nonlinear function of plasma properties (e.g. eqs 110-113 in [1]) makes a fully nonlinear solve arguably difficult in the first instance, bearing in mind the likely time constraints of the development.
- For implicit-in-time problems, much of the infrastructure around *Nektar++* is designed for a series of scalar solutions vs. the solution of a large coupled system. For example in the incompressible Navier-Stokes equations the use of the velocity correction scheme of [4] involves the solution of a pressure Poisson and three Helmholtz equations for velocity. There are however exceptions to this, as we discuss in the below.

However, given an initial explicit-in-time implementation, there is the potential to consider implicit implementations based on the techniques proposed in [5]. This work has extended the existing explicit implementation of the compressible Navier-Stokes equations through the Jacobi-free Newton-Krylov method (JFNK). In this approach, the explicit solver is evaluated twice in order to determine an approximate Jacobian based on a simple first-order approximation, and the framework in *Nektar++* has been designed to be reasonably generic in its use for other systems. Theoretically the JFNK approach enables the implicit solution of any system, given an appropriate explicit solver; however, in practical terms, to address conditioning issues in the solution of the resultant system, it is often necessary to precondition the system by using appropriate parts of the linearised equations. Such an approach could be adopted here, although deriving appropriate preconditioner terms and their implementation complexity would likely be considerable.

4.3 Particle coupling

As noted in [1], there is no explicit coupling of this system with particle models within the interior of the domain. However, the sheath boundary condition is more accurately represented as particles. Appropriate interfaces must therefore be made in order to enable this integration. These are presently the subject of discussion between UKAEA and the authors, and conceptually the process has been established to e.g. determine cell information and query for solution values given a position in space; however more detailed technical development is required to establish the parallel execution strategy.

4.4 Practical considerations

System 2-5 as stated is highly complex involving not only a large number of parameters, variables and equations, but a large amount of domain-specific knowledge in terms of the underlying physics. A successful solver development will therefore require a balance of efforts between an interdisciplinary team comprising experts in the numerical implementation as well as assistance in interpretation and translation from the plasma physics & fusion domains. For example:

- Initial assistance will be required in developing the appropriate solution pathway and time integration scheme in a more complete form. The lack of a DSL within *Nektar++* necessitates more direct implementation, and is perhaps an area for future development (e.g. with the provision of a lightweight DSL in the setting of an explicit solver, the specification of these equations would be more straightforward).
- Some terms (e.g. τ. terms, eqs 110-113 in [1]) are written as either constants or nonlinear functions of the plasma; guidance as to the most appropriate route as the code is developed is critical.
- Even after initial development, instabilities and errors will be inevitable: tracking down the source of these instabilities and determining their nature (as either implementation-specific or physics-based) requires input from both 'teams' in this setting.

5 Conclusions

This deliverable has briefly outlined the likely development route for a solver based around system 2-6. The final phase of this project will be to start in the development of some aspects of this solver; for example in discussion with partners at York and UKAEA, to map out the solver structure and order of operations given the system. This progress will be reported in the final deliverable, D5.3 at the conclusion of this project, along with any initial development efforts that are possible.

References

[1] Wayne Arter. Equations for NEPTUNE Proxyapps. Technical report, UK Atomic Energy Authority, 2020.

[2] Kambiz Salari and Patrick Knupp. Code verification by the method of manufactured solutions. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2000.

[3] Benjamin Daniel Dudson, Jens Madsen, John Omotani, Peter Hill, Luke Easy, and Michael Løiten. Verification of BOUT++ by the method of manufactured solutions. *Physics of Plasmas*, 23(6):062303, 2016.

[4] George Em Karniadakis, Moshe Israeli, and Steven A Orszag. High-order splitting methods for the incompressible navier-stokes equations. *Journal of computational physics*, 97(2):414–443, 1991.

[5] Z. Yan, Y. Pan, G. Castiglioni, K. Hillewaert, J. Peiró, D. Moxey, and S. J. Sherwin. *Nektar++*: Design and implementation of an implicit spectral/hp element compressible flow solver using a jacobian-free newton krylov approach. *Comput. Math. Appl.*, 81:351–372, 2021.